Rapid Spacecraft Payload Development: In-Orbit Demonstration of Flight Software Reuse, Scalability, and Dependability

Walter Vaughan[*†§], Caleb Adams[†], Sergei Gridnev[†], Nick Cramer[†],
Alice Anlind[‡], Eric Brune[‡], Oskar Flordal[‡], Fredrik Bruhn[‡], and Alan George[*]

[*] NSF Center for Space, High-Performance, and Resilient Computing, University of Pittsburgh, Pittsburgh, PA
[†] Intelligent Systems Division, NASA Ames Research Center, Moffett Field, CA
[‡] Unibap AB, Uppsala, Sweden
[§] wbv1@pitt.edu

Abstract

As space mission design trends towards shared, multi-mission platforms and high-performance onboard computing architectures, the number of spacecraft launched into operation is also steadily rising. Through ridesharing, spacecraft miniaturization, and other cost-reduction measures, the barriers to space are lowering, resulting in compounded growth in the amount of flight software being deployed. To meet the needs of both the growing quantity and evolving nature of spacecraft, flight software design must accordingly adapt to support more efficient development, solutions to computational resource-sharing, and software reusability. This paper focuses on a software payload demonstrating several core technologies that improve the state-of-the-art in these identified areas. Launched into low-earth orbit in January 2022, our software payload was conceived, designed, and delivered in a span of merely two months. It was developed on top of the NASA core Flight System (cFS) framework and the Distributed Spacecraft Autonomy (DSA) Comm cFS application, which translates cFS software bus messages across a Data Distribution Service (DDS) network. The flight software, packaged in Linux container images, was deployed as one of 18 flight applications managed through the Unibap SpaceCloud Framework. The applications were run on a Unibap iX5-102 radiation-tolerant payload computer, hosted on the D-Orbit SCV-004 spacecraft as part of an ESA-sponsored in-orbit technology test.

Our payload, referred to as the *DSA D-Orbit software*, demonstrates the reusability of the DSA Comm app in a substantially different context and purpose as its original mission. Comm's original design goal was to reliably distribute messages between spacecraft swarms of arbitrary size and dynamic network topology. However, we leverage this same functionality to introduce redundancy and opportunistic parallel data processing in the context of a representative onboard image processing workload. This adaptive mission architecture was enabled in part by the SpaceCloud Framework's use of container virtualization as the payload integration interface. By using a base container image with common high-level language runtimes and libraries, we were able to rapidly design, develop, and validate our image processing application without many of the technological barriers common to flight software development. We present details the goals, approach, results, and lessons learned through this technology demonstration experiment and contextualize those observations against present and future challenges in spacecraft software development.